ATTORNEY DOCKET NO. 4256

APPENDIX A

for

METHOD AND APPARATUS FOR INDEXING, SEARCHING AND DISPLAYING DATA

Libertech, Inc.

V-Search™ Integration Toolkit for Folio VIEWS Beta Release 2.0

User's Manual

PRELIMINARY DRAFT

Draft 1.0

6 December, 1995

V-Search Integration Toolkit for Folio VIEWS

User's Manual

V-Search Integration Toolkit for Folio VIEWS Beta Test Edition - Beta Release 2.0.

These materials apply to Release 2.0 of the V-Search Integration Toolkit for Folio VIEWS, to be circulated for Beta Testing only.

In keeping with the Beta Test Agreement, these materials may not be copied or reproduced in any form without written permission of Libertech, Inc. Requests for permission to reproduce these materials in whole or in part, or to receive multiple sets on a regular basis, should be addressed to:

Christopher J. Uzzi
Director of Product Services
Libertech, Inc.
2200 West Main Street, Suite 230B
Durham, NC 27705

Trademarks

Preface

This manual shows you how to use the V-Search Integration Toolkit for Folio VIEWS to combine the text-handling capability of Folio VIEWS with the searching and navigation functions of V-Search. This lets you use the V-Search Viewer for Folio VIEWS as an extension to the Folio VIEWS Infobase Manager to provide your users with the ability to:

- Jump from documents in a Folio VIEWS infobase to a V-Search Map.
- Jump from a V-Search Map to documents in a Folio VIEWS infobase.

Who Should Read this Manual?

You should read this manual if you will integrate a V-Search database with a Folio VIEWS infobase.

How is this Manual Organized?

"Chapter 1: Getting Started" on page 1 presents an overview of the interaction between the V-Search Viewer and Folio VIEWS Infobase Manager, and describes how to install the toolkit.

"Chapter 2: Creating the Folio VIEWS Integration Table" on page 7 describes how to build a table, called the V-Search Folio VIEWS Integration Table, that makes it possible to retrieve documents contained in a Folio VIEWS infobase from a V-Search Map.

"Chapter 3: Compiling the Folio VIEWS Integration Table" on page 13 describes how to compile the integration table.

"Chapter 4: Creating Jumps From Folio VIEWS " on page 20 describes how to add program links to the text of a Folio VIEWS infobase so that users can jump to V-Search Maps from documents in the infobase.

"Chapter 5: Marking Up Your Folio Flat Files for V-Search" on page 23 describes several possible ways you can use Folio flat file tags to improve the way your infobase works with V-Search.

"Chapter 6: Completing the Package" on page 26 lists the files and libraries to include when you package the V-Search database and viewer with Folio VIEWS on a CD-ROM, and gives you a brief description of how the integration process works.

What Should You Know Already?

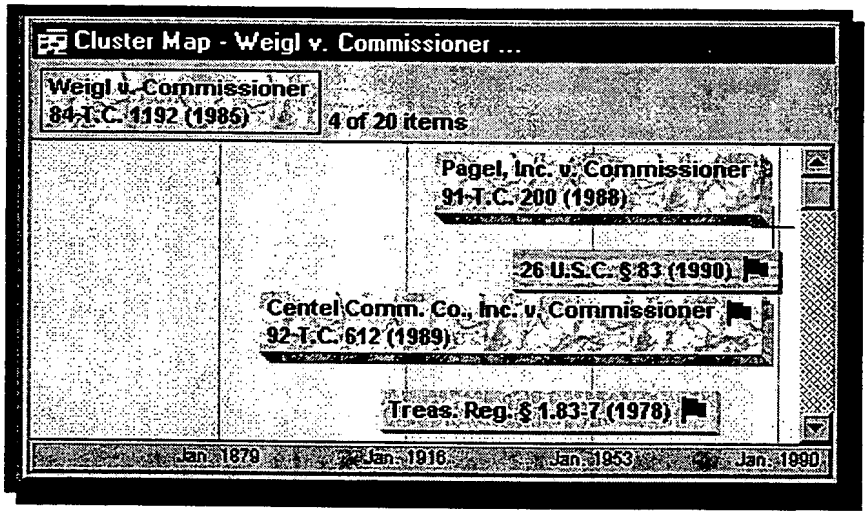
Before you start to use this toolkit, you should have experience in:

- Using the Folio VIEWS Infobase Manager to view an infobase.
- Preparing Folio VIEWS flat files (*.FFF) for generating infobases.
- Developing and using Folio VIEWS query and jump commands.
- Defining nodes and links in a V-Search database, as described in the V-Search Publisher's Toolkit User's Manual.

Software Prerequisites

- The V-Search Publisher's Toolkit and its prerequisites.
- The Folio VIEWS 3.1 Professional Infobase Development Kit.

Contents



Frontispiece: Sample V-Search Map.

Chapter 1: Getting Started

This manual describes how to combine the functions of V-Search and Folio VIEWS so that your users will be able to display V-Search Maps from VIEWS, and display infobase text from V-Search Maps. Your users will be able to use all the power of the V-Search Maps to search for relationships among the documents in an infobase.

Methods of Integration

There are two ways in which you can integrate the V-Search database and VIEWS infobases.

- You can integrate the V-Search Map node boxes with text documents in an infobase. This lets a user view the underlying text from a node box on a V-Search Source, Influence, or Cluster Map.
- You can integrate documents in the infobase text with V-Search Maps. This lets a user jump to a V-Search Map directly from program links in the infobase text.

Jumping From the Maps to the Text

You can integrate the V-Search Maps with the infobase text, so that a user can jump directly from a node on a map to the text that is represented by that node. Here's how it happens.

Before ...

This is what happens ***before*** you integrate V-Search Maps with the VIEWS infobase.

When a user clicks on a node box on a V-Search Map, a menu lets the user choose whether to display a Source, Influence, or Cluster Map for that node. However, the user cannot see the text of the document that is represented by that node. In Figure 1, a user has clicked the node box for 26 U.S.C. § 83. Note that the menu options are limited to V-Search maps.

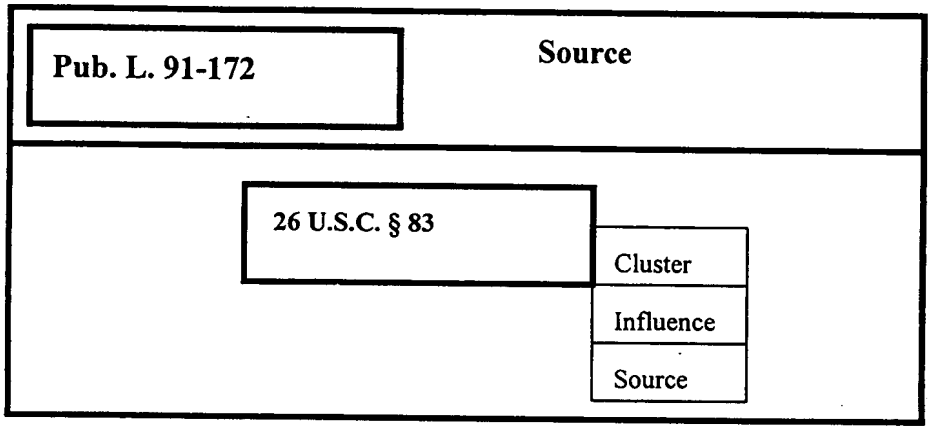


Figure 1. V-Search Map with Menu.

After ...

This is what happens ***after*** you integrate the V-Search database and the infobase.

When the user clicks on a node box on a V-Search Map now, the menu lets the user view the text underlying the node, or the text of the search node that refers to this result node, in addition to the V-Search Maps. Here, the user clicked on the node box for 26 U.S.C. § 83. Notice the additional selections on the menu. When the user then clicks on the Show Text or Show Usage command, the Folio VIEWS Infobase manager will display the text of a node. Figure 2 shows what happens when the user selects Show Text to jump from the V-Search Map to the VIEWS text.

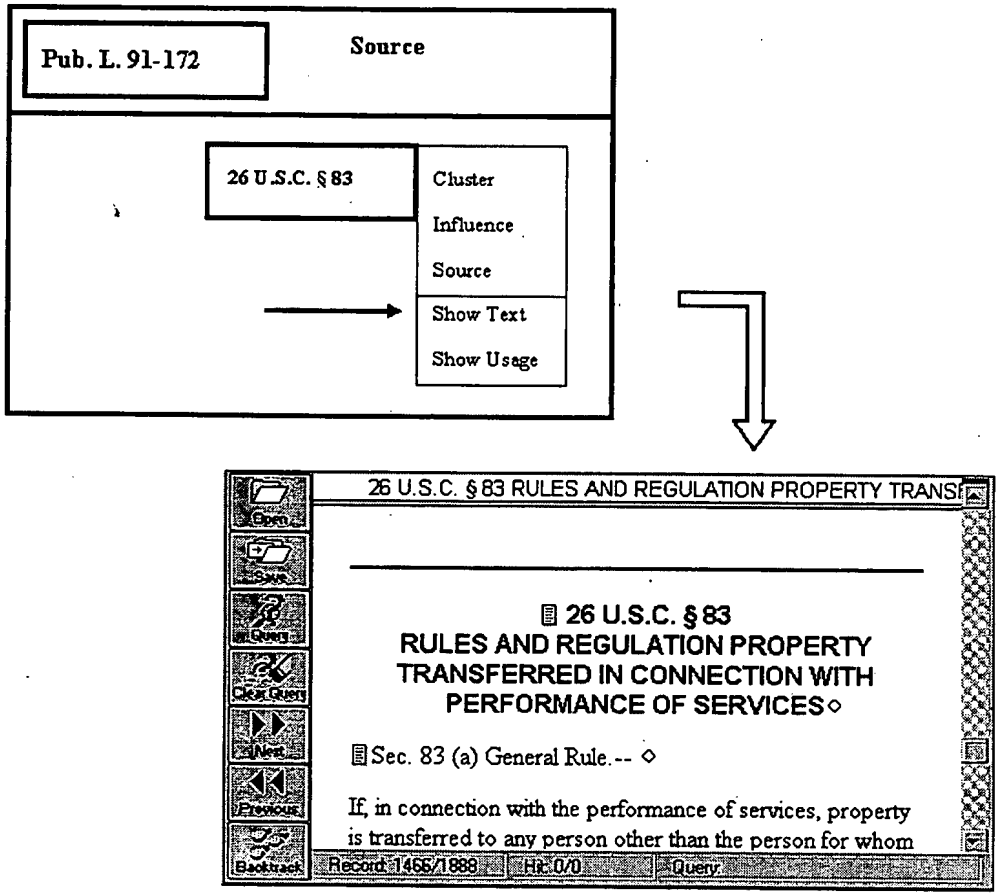


Figure 2. Show Text Jumps from a V-Search Map to the VIEWS Text.

The Show Usage and Show Text Commands

The V-Search Integration Toolkit for Folio VIEWS adds the Show Text and Show Usage commands to the menu for the nodes on a V-Search Map.

Show Text – The Show Text command lets the user see the text of the document that is represented by the selected node. The node can be either the search node or a result node. In either case, the infobase text is displayed for that node. If the user selects Show Text for a result node, the text of the result node is shown with any references to the search node highlighted.

Show Usage – The Show Usage command lets the user see the text of the document that is represented by the search node. This is available only from a result node. When the user selects Show Usage from a result node, the text for the search node is displayed showing the references to the result node highlighted.

How You Make it Happen ...

To integrate the V-Search Map with the infobase text, you:

1. Mark up your Folio flat files to provide tags used by V-Search queries, as shown in Chapter 5.
2. Create a table in the V-Search database to associate the nodes in the Node table with documents in an infobase, as shown in Chapter 2. You will need one integration table for each infobase tied to your V-Search database. This table, called the Folio VIEWS Integration Table (or just the integration table) associates nodes in the Node table with VIEWS Jump or Query commands.
3. Compile the integration table, as shown in Chapter 3, so that it can be distributed with the database and infobase.

Jumping From the Text to the Maps

You can integrate the infobase text with V-Search Maps so that a user can jump directly from the infobase text of a document to a map that shows the links to and from the node that represents that document. Here's how it happens.

Before ...

This is what happens ***before*** you integrate the text in the infobase with V-Search.

When the user views text in the infobase, as shown in Figure 3, there is no way to jump to the V-Search Maps from the text.

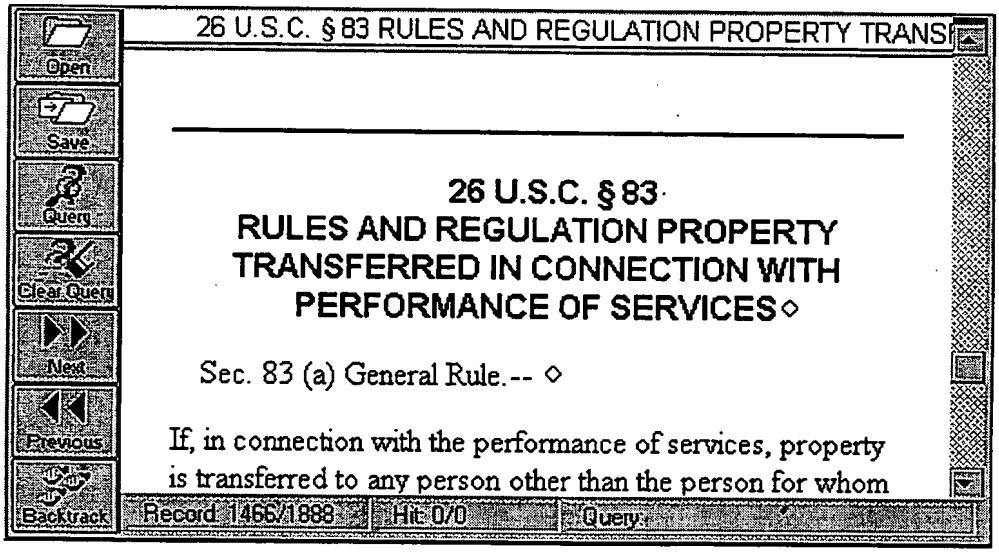


Figure 3. VIEWS Text With No Jumps.

After ...

This is what happens ***after*** you integrate the text in the infobase with V-Search.

You define program links in the text. When the user views the text with VIEWS, the program links provide access to V-Search Maps. Figure 4 shows the entire process, starting in the VIEWS text with a program link, and the jumping to a search dialog that displays a V-Search Map.

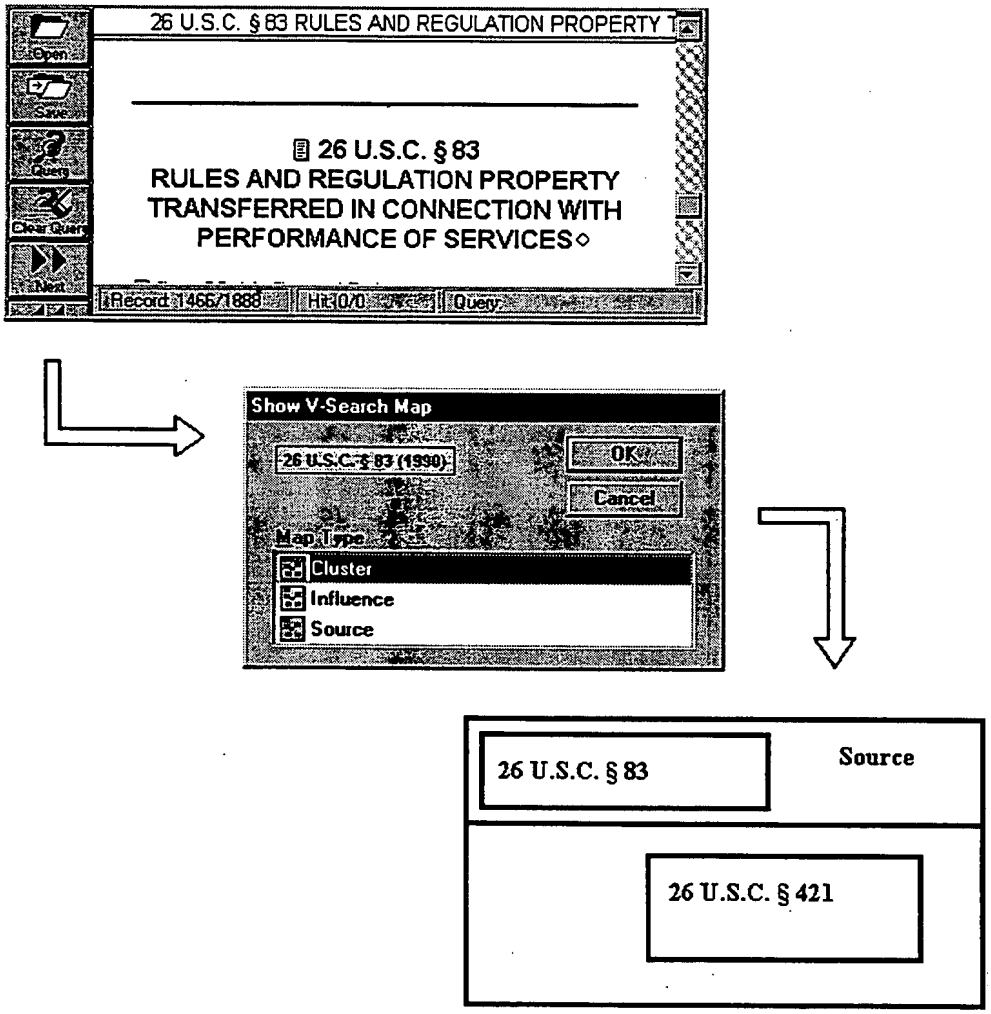


Figure 4. Jumping From the Text to a Map.

How You Make it Happen ...

To integrate the infobase text with the V-Search Map, you add program links to the text, as shown in Chapter 4. When activated, each program link displays a dialog that lets the user access V-Search Maps. For each citation in the infobase text, you add a VIEWS program link. The program link calls a V-Search program to display a dialog box that prompts the user to select a map.

Installing the V-Search Folio VIEWS Toolkit

When you are ready to install the V-Search Integration Toolkit for Folio VIEWS, insert the installation CD in the appropriate CD-ROM drive. This section tells you how to install the toolkit from the installation CD to a disk on your computer.

1. Start Microsoft Windows NT or Windows 95 on your computer.
2. Insert the installation CD in the CD-ROM drive.
3. In Windows NT, select the File option from the upper left corner of the Program Manager window. From the File pulldown menu, select Run.

   In Windows 95, select the Run item from the Start Menu.
4. In the Run command line box, enter:

   drive: setup

   Where *drive* is the letter of the drive where you inserted the installation CD.

The V-Search Integration Toolkit for Folio VIEWS Setup program starts.

Installed Components

When you install this toolkit, these files and libraries are installed on your computer:

File	Description
VSRCHFVE.DLL	V-Search Viewer for Folio VIEWS VSRCHFVE.DLL is the 16-bit V-Search Extension to Folio VIEWS. This library provides for displaying V-Search Maps from a Folio VIEWS Infobase manager.
FOLIOVWS.VXT	V-Search Extension Library for Folio VIEWS (16-bit version) FOLIOVWS.VXT is the 16-bit Folio VIEWS V-Search Extension Library. This library is used for displaying Folio VIEWS text from a V-Search Map within the V-Search Viewer for Folio VIEWS (VSRCHFVE.DLL).
VSFVCOMP.EXE	V-Search Folio VIEWS Integration Table Compiler VSFVCOMP.EXE compiles the V-Search Folio VIEWS Integration Table.
VSFVJUMP.EXE	V-Search Folio VIEWS Program Jump Executor VSFVJUMP.EXE notifies the V-Search Viewer to display a search dialog for the selected node. This is used in program links to jump from VIEWS text to a V-Search Map.
LINK.BMP	LINK.BMP is a bitmap that you can use to mark a Program Link from a citation to a V-Search Map.
NODE.BMP	NODE.BMP is a bitmap that you can use to mark a Program Link from a document to a V-Search Map.

Integration Summary

To complete the integration of your compiled V-Search database with Folio VIEWS:

1. Install the V-Search Folio VIEWS Toolkit on the same computer with the V-Search Publisher's Toolkit. See "Installing the V-Search Folio VIEWS Toolkit" on page 5 for instructions.
2. Mark up your Folio flat files with tags that can improve integration with V-Search. See "Chapter 5: Marking Up Your Folio Flat Files for V-Search" on page 23 for suggested ways to mark up your files.
3. Create a Folio VIEWS Integration Table (called the integration table) to associate nodes from the V-Search Node table with VIEWS Jump and Query commands. See "Chapter 2: Creating the Folio VIEWS Integration Table" on page 7 for instructions on building an integration table.
4. Compile the Folio VIEWS Integration Table. See "Chapter 3: Compiling the Folio VIEWS Integration Table" on page 13 for instructions on compiling your integration tables to prepare for distribution.
5. Create program links in the Folio flat files to display V-Search Maps. See "Chapter 4: Creating Jumps From Folio VIEWS " on page 20 for instructions on creating program links.
6. Package the V-Search database, and the V-Search Viewer with its libraries, with the VIEWS infobase and Infobase Manager for shipment on a CD-ROM, as described in "Chapter 6: Completing the Package" on page 26.

Chapter 2: Creating the Folio VIEWS Integration Table

When a user clicks on a node box on a V-Search Map, a menu is displayed that lets the user select other V-Search Maps to view. When you install the V-Search Folio VIEWS Extension, this menu also lets the user view the text of the document that is associated with the node, and the text of the document that refers to the node. To provide the relationships between nodes in the Node table and text in the infobase, you create an additional database table called a Folio VIEWS Integration Table. This integration table associates nodes in the Node table with Folio VIEWS commands that display the text of a document from an infobase.

This chapter tells you how to build the integration table. Use your database manager to create this table in the V-Search database. Create one integration table for each infobase represented in your V-Search database. Each integration table relates the documents in one specific infobase to a set of nodes in the Node table.

Table Name

Each integration table must have the same name as the Folio VIEWS infobase with which it is used. For example, if your infobase is named CLASS1.NFO, you would create an integration table named CLASS1 in the V-Search database.

Table Structure

This section shows the structure of the integration table:

Column Name	Data Type	Data Length	Key Value
NodeID	text	60	1
TextLength	number	4	
ShowTextMethod	text	5	
ShowTextCmd	text	1-255*	
ShowUsageQuery	text	1-255*	

* You can define these column lengths from 1 to 255 characters, depending on the size of the ShowTextCmd and ShowUsageQuery commands you use.

Column Contents

This section describes the purpose and content of each column in the table. It tells you the name of each column, and what you can enter into that column. Each row provides information for one NodeID.

NodeID

This column can contain:

- The unique NodeID of a node as defined in the Node table in the V-Search database. The NodeID you enter here must match a NodeID in the Node table.

- A unique AliasID that defines a Folio VIEWS command, with substitutable options, that can be referred to by other entries in this table. See "Using Aliases in NodeID Entries" on page 9 for more information.

TextLength

This column contains the length of the text that is associated with this node. The V-Search Viewer uses this value to determine the depth of a node box relative to the other node boxes on a V-Search Map. If this value is 0, the node box representing this node will be flat; the Show Text command will be disabled because there is no text to display.

You can use any unit of measure for the TextLength, but you should use a value that is consistent across the database. For example, you could use the number of characters, words, or pages in the document as its TextLength value.

Note: For consistency among V-Search Maps produced by different publishers, we suggest that you use the approximate number of words in the text as the standard measure for the TextLength value.

ShowTextMethod

This column contains a text string that specifies whether the ShowTextCmd column for this entry contains a Folio VIEWS **Query** command or **Jump** Link destination. The valid entries are:

Value	*Meaning*
Query	The ShowTextCmd column contains a Query command.
Jump	The ShowTextCmd column contains a *destination* for a Jump Link command.

Note: If the ShowTextCmd command is an Alias (as described in "Using Symbols in Commands" on page 9), you need not enter a value for the ShowTextMethod. Instead, the ShowTextCmd command will use the ShowTextMethod associated with the AliasID entry.

ShowTextCmd

This column contains the Folio VIEWS command to display document text for the Show Text option. This column can contain one of the following:

- A Folio VIEWS Query command, that will be run to show the document text, if the ShowTextMethod column contains Query.
- The name of a jump destination in the infobase, if the ShowTextMethod column contains Jump. VIEWS executes a Jump Link command to display the infobase text beginning at the specified Jump Destination tag.
- An AliasID, enclosed in # characters, to execute the command in the ShowTextCmd column of the AliasID entry.

The Show Text command is enabled only if the TextLength value is greater than 0 (there is text to view).

Note: Using AliasIDs as the actual ShowTextCmd and ShowUsageQuery commands can save you a lot of time, and can provide a consistent behavior for your maps.

ShowUsageQuery

This column contains the Folio VIEWS Query command to display document text for the Show Usage option. The command in the ShowUsageQuery column should be a VIEWS query that displays the text of the search node with the references to the result node highlighted.

This column can also contain an AliasID, enclosed in # characters, to run the command in the ShowUsageQuery column of the AliasID entry.

The Show Usage command is added to the menu when there is at least one node in the integration table that has a command in the ShowUsageQuery column. If you do not want to implement this function, be sure that the ShowUsageQuery column is empty for all entries in the integration table.

The Show Usage command is enabled for a result node if there is a ShowUsageQuery command in the integration table entry for the NodeID of the search node.

Using Symbols in Commands

You can imbed symbolic values in the commands that you enter in the ShowTextCmd and ShowUsageQuery columns. There are two types of symbols: substitutable values and aliases.

Using Substitutable Values

A substitutable value is a replaceable parameter that you can use in the commands in the ShowTextCmd and ShowUsageQuery columns of the integration table. Enter the substitutable value name enclosed in a pair of # characters. When the end user selects Show Text or Show Usage from a map, the substitutable value in the ShowTextCmd or ShowUsageQuery command is replaced by the NodeID whose text is to be displayed. You can use these two substitutable values in the integration table:

Substitutable Value	Use
#scope#	Is always replaced by the NodeID of the node whose text is displayed.
#reference#	Is replaced by the NodeID of the node whose references in the #scope# node you want to highlight.

When the Folio VIEWS command is run, the occurrences of #scope# are replaced by the NodeID of the document whose text you are viewing. The occurrences of #reference# are replaced by the NodeID of the document whose references you want to highlight in the text of the #scope# node.

Using Aliases in NodeID Entries

In the section "NodeID" on page 7, you learned that you can define an integration table entry with either a real NodeID or an AliasID. An AliasID entry contains commands that can be referred to by the ShowTextCmd and ShowUsageQuery columns of the NodeID entries. Using AliasIDs as commands can save you a lot of time, and can provide a consistent behavior for your maps.

For example, if all of the NodeIDs in the table use the same ShowTextCmd command format, you can create an AliasID entry that has the actual ShowTextCmd command text. Then, in all the other entries in the integration table, you can refer to that AliasID. The syntax of an alias is:

```
#@NodeID_Value#
```

Replace *NodeID_Value* with the AliasID of the entry that contains the command to be run. Here is an example of an alias and the command to which it refers:

NodeID	Text Length	Show Text Method	ShowTextCmd	ShowUsageQuery
92 T.C. 612	4098		#@cases#	#@cases#
cases	0	Query	[group "#scope#":*]	[group "#scope#": [field "#reference#": *]]

In the first entry, the alias #@cases# refers to the AliasID in the second entry. Whenever a Show Text or Show Usage is run for the node "92 T.C. 612," the command to show the text is taken from the cases entry in the table.

Note: An alias must be the only item in the ShowTextCmd or ShowUsageQuery column for an entry; you cannot imbed an alias in another command.

Displaying Text with Show Text and Show Usage

This section describes what happens when the end user selects the Show Text or Show Usage menu item from a node box menu on a V-Search Map.

How Show Usage Works

The Show Usage menu item is available only for a result node. When an end user selects Show Usage, the V-Search Viewer:

1. Determines the NodeID of the search node.
2. Determines whether this entry is a command or an alias.
   - If it is a command, runs the VIEWS query command from that entry using:

     The search node as the #scope# of the query, and
     The result node as the #reference#
   - If it is an alias, runs the VIEWS query command from the AliasID entry.

How Show Text Works

The Show Text process is more complicated. It depends on whether the ShowTextMethod is a Jump or a Query. When the end user selects Show Text, The V-Search Viewer:

1. Determines the NodeID of the node from which the user selected Show Text.
2. Looks in the integration table for the NodeID entry for that node.
3. Runs the ShowUsageQuery query command from that entry to highlight all the references in the text, using:

   The result node as the #scope# of the query, and
   The search node as the #reference#

   And positions the viewer at the first reference.
4. Checks whether there were any references to highlight:

   a) If there are **no** references to highlight, executes the ShowTextCmd command.

   b) If there **are** references to highlight, determines whether the ShowTextMethod is a Jump or a Query.

   - If ShowTextMethod is a Jump, executes the Jump to the Jump Destination at the beginning of the text.
   - If ShowTextMethod is a Query, stop; does NOT execute the ShowTextCmd query because it would destroy the results of the ShowUsageQuery command just run. This leaves the viewer positioned at the first reference in the text.

If you use ShowTextMethod = Jump, the viewer is always positioned at the beginning of the text.

If you use ShowTextMethod = Query, the viewer is sometimes positioned at the first reference in the text rather than at the beginning.

Example: Markup Using Group and Field Tags

The Folio VIEWS commands that you put in the integration table are determined by the tags that you use in the Folio flat files to mark citations. This section shows a sample integration table and a sample of the Folio flat file markup with which it is used. The Folio flat file markup that uses:

- A Jump Destination tag to mark the start of the text for each node.
- Group tags to mark all the records (paragraphs) that contain text for each node.
- Field tags to provide a standard way to mark references to other nodes.

It then shows a sample integration table based on using those tags. First, here is the text and markup:

```
<RD><GR:"92 T.C. 612">
<JD: "92 T.C. 612">
(1) Whether the stock warrants were issued "in connection with the
performance of services" within the meaning of
<JL:Goto,"26 U.S.C. § 83">
<FD:"26 U.S.C. § 83">
section 83
</FD:"26 U.S.C. § 83"></JL>
```

Notice the <GR> tag marks this record as a part of the text for document 92 T.C. 612. Then the <FD> tag marks a reference to document 26 U.S.C. § 83. In the integration table, the queries will use:

group	To search for the text of the document to display, and to specify the scope of a reference query.
field	To highlight citations in that text.

NodeID	Text Length	Show Text Method	ShowTextCmd	ShowUsageQuery
92 T.C. 612	4098		#@cases#	#@cases#
26 U.S.C. § 83	2343		#@statutes#	#@statutes#
cases	0	Query	[group "#scope#":*]	[group "#scope#": [field "#reference#": *]]
statutes	0	Jump	#scope#	[group "#scope#": [field "#reference#": *]]

Chapter 3: Compiling the Folio VIEWS Integration Table

This chapter describes how to compile the integration table for shipment with an infobase. The Folio VIEWS Integration Table Compiler is VSFVCOMP.EXE.

Using the Compiler

You can specify the options for the compiler on the command-line or interactively in a dialog box. The only required parameters are the /Database and /Table options, which specify the names of the V-Search database and integration table to be processed. If you do not include these options on the command line, a startup dialog box is displayed to let you enter the options.

Command-Line Interface

Options on the command line are processed from left to right. If you specify an option more than once, the rightmost setting is used. Parameter files are processed in the order that they are found on the command line. Options to the right of a /ParamFile option on the command line will override any settings in the parameter file.

If you enter an unknown or invalid option, a warning message is generated.

If you enter a correct option but an invalid value for that option, a warning message is displayed and the default value is used for that option.

In the following syntax diagram:

- Items shown in *italics* are variables; you supply an appropriate value.
- Items shown in [brackets] are optional.
- An underlined item is a default value.
- The options are **not** case sensitive.

The command-line syntax for VSFVCOMP is:

```
VSFVCOMP   [/ParamFile=filename]
           [[/Database=]databasename]
           [/Table=tablename]
           [/Warnings=[yes|no]]
           [/ReportTo=[dialog|file]]
           [/LogFile=filename]
           [/LogOption=[append|overwrite]]
           [/Progress=[yes|no]]
           [/EstimateTime=[yes|no]]
           [/UpdateInterval=[0|1|seconds-between-updates]]
           [/StartupDialog=[yes|no]]
           [/Target=filename]
           [/Overwrite=[yes|no]]
```

/ParamFile=*filename*

Use this option to specify a file that contains options for VSVFCOMP. You can specify the options in a parameter file as well as on the command line. When you specify this /ParamFile option alone, all options are taken from the parameter file. When you specify this /ParamFile option with other options on the command line, the options in the file are used with the options on the command line. Options to the right of a /ParamFile option on the command line override equivalent settings in the parameter file.

Sample Parameter File: Here is a sample parameter file:

```
Database=VSEARCH.VDB
Target=Vsdemo.FTB
Table=Vsdemo
```

Note that the / character is not used for the options in this sample parameter file. The / character is optional in a parameter file. Any line in the parameter file that begins with either a / or an alphabetic character is treated as an option. Any line in the parameter file that does **not** begin with a / or an alphabetic character is ignored.

The /ParamFile option is itself optional. If you do not specify /ParamFile on the command line, no parameter file is used.

/Database=*databasename*

Use this option to specify the name of the V-Search database to be processed. Specify the file name of a V-Search database whose file extension is VDB. If you omit this option on the command line, a dialog box is displayed as shown in "Interactive Interface" on page 17. Enter the information in the dialog box and press OK.

Note: The parameter name /Database is itself optional. Instead, you can start the compiler, without the "/Database=" syntax, by typing just:

```
VSFVCOMP databasename
```

/Table=*tablename*

Use this option to specify the name of the Folio VIEWS Integration Table to compile. The /Table option is case-sensitive. This name is **required.** If you omit this option on the command line, a dialog box is displayed as shown in "Interactive Interface" on page 17. Enter the integration table name and press OK.

/Warnings=[yes|no]

Use this option to specify whether this program will generate warning messages as well as error messages. Enter /Warnings=yes to display warning messages. Enter /Warnings=no to suppress warning messages. If you omit this option, warning messages are displayed.

/ReportTo=[dialog|file]

Use this option to specify whether this program displays messages in a dialog box or sends messages to a log file. Enter /ReportTo=dialog to display messages in a dialog box. Enter /ReportTo=file to send messages to the log file specified by the /LogFile option. If you omit this option, messages are displayed in a dialog box.

Displaying Messages in a Dialog Box: When a warning message is displayed in a dialog box, there are OK and Cancel buttons available. Push OK to continue processing. Push Cancel to end the program. While a dialog box is displayed, the process is halted until you press a button.

When an error message is displayed in a dialog box, only the OK button is displayed. Press OK to end the program.

Storing Messages in a Log File: If you choose to store messages in a log file, warning messages are recorded but the program continues to run. However, an error message will end the process.

/LogFile=*filename*

Use this option to specify the name of a log file in which to store error and warning messages that you do not want to display in a dialog box. If you omit this option, but specify /ReportTo=file, messages are stored in the default log file in the same directory as the input VDB file, with the file name Integration Table Compiler Log.TXT.

/LogOption=[append|overwrite]

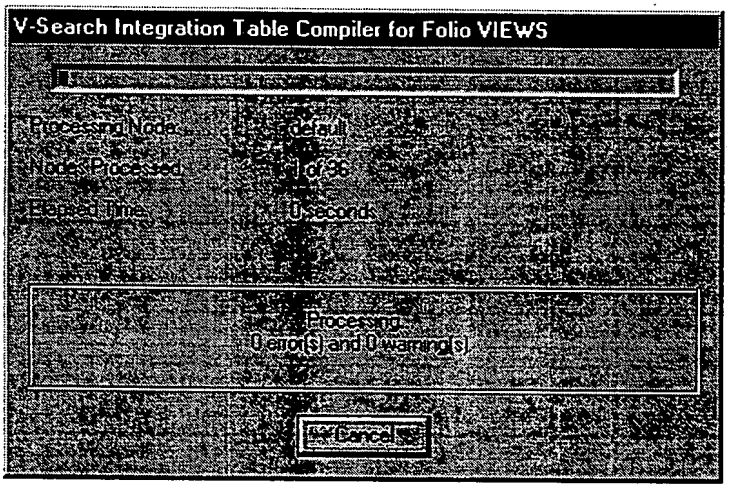
Use this option to specify whether messages that are stored in a log file are appended to the end of the file or replace the contents of the file. Enter /LogOption=append to add messages to the end of the log file. Enter /LogOption=overwrite to replace the previous contents of the file with the new messages as they accumulate.

/Progress=[yes|no]

Use this option to specify whether the program displays a progress indicator. The progress indicator tells you how many items have been processed. Enter /Progress=yes to display a progress indicator as the program runs. Enter /Progress=no to omit the progress indicator. If you omit this option, /Progress=yes is used.

Performance Note: For some database systems, the process of counting the items can severely inhibit the processing of the database. Using /Progress=No will improve the performance of the compiler.

Following is a sample progress indicator:

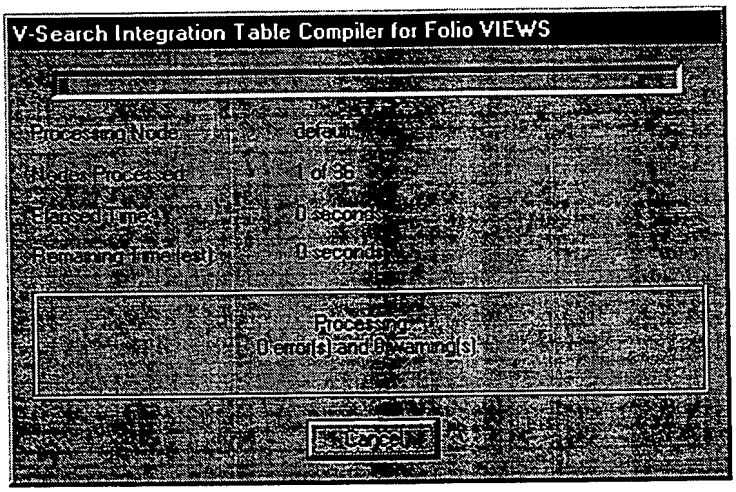


The Cancel button on the progress indicator lets you stop the process. A dialog box will then prompt you to confirm this action before ending the program.

/EstimateTime=[yes|no]

Use this option to specify whether the progress indicator also displays how many items remain to be processed in the database. Initially, the total number of items to be processed is not known. After the total number of items has been determined, if you enter /EstimateTime=yes, the progress indicator also displays:

- The number of items completed.
- The total number of items in the database.
- The current item.
- The elapsed time.
- An estimate of the remaining time. This estimate is based on the average time to process each item.



/UpdateInterval=[0| 1 | *seconds-between-updates*]

Use this option to specify the number of seconds between updates of the progress indicator. If you omit this option, the progress indicator is updated at 1-second intervals. The special value 0 updates the progress indicator as each node is processed by the tool. If you select 0 and the tool processes 5 nodes per second, the progress indicator is updated 5 times per second too.

/StartupDialog=[yes|no]

Use this option to specify whether the startup dialog box is displayed. By default, this program displays a startup dialog box only if you omit the /Database or /Table options on the command line. The /StartupDialog option lets you start the program from the command line, but enter options in a dialog box. The startup dialog box is illustrated in "Interactive Interface" on page 17. Enter the information in the dialog box and press OK to start the process.

/Target=*filename*

Use this option to specify the name of the compiled integration table. If you omit this option, the compiled output is stored in a file with:

- The same path as the path to the VDB file.
- The same file name as specified for the /Table option.
- A file extension of FTB.

If the resulting output filename already exists, you are prompted to confirm that it can be overwritten. You can use the /Overwrite option to specify an automatic overwrite of an existing file.

/Overwrite=[yes|no]

Use this option to specify whether to automatically overwrite the named output file if it already exists. Enter /Overwrite=yes to automatically overwrite an existing file. Enter /Overwrite=no if you do not want to automatically overwrite an existing file. If you omit this option, /Overwrite=no is used.

Interactive Interface

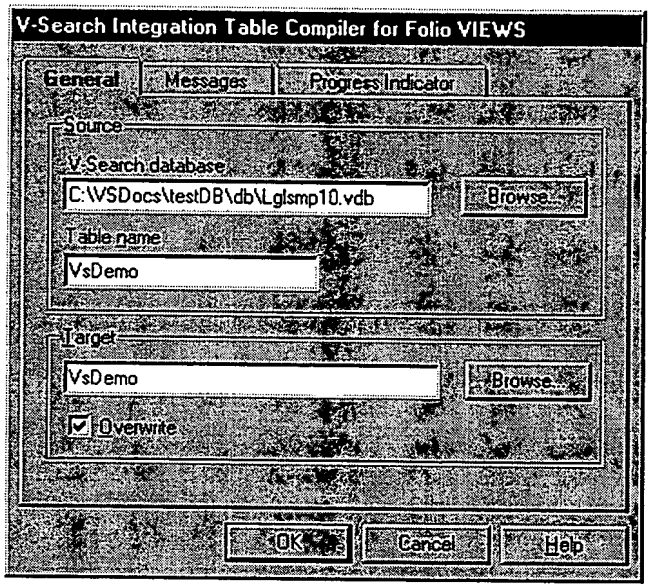
A startup dialog is displayed when you do not specify the /Table and /Database options on the command line, or when you enter the /StartupDialog=yes option. This allows you to specify the options in a dialog box. All options are as described in the "Command-Line Interface " section.

Complete the tabs in the dialog and press OK to begin compiling.

General Tab

Use the General tab to enter these options:

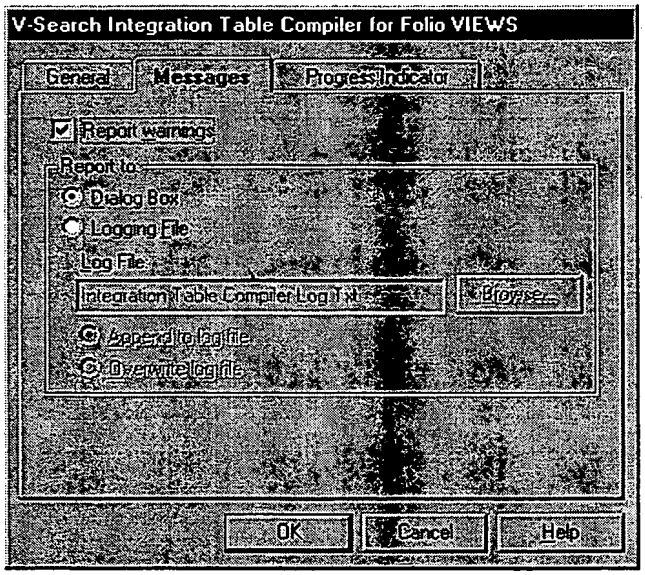
/Database=*databasename*
/Table=*tablename*
/Target=*filename*
/Overwrite=yes|no



Messages Tab

Use the Messages tab to enter these options:

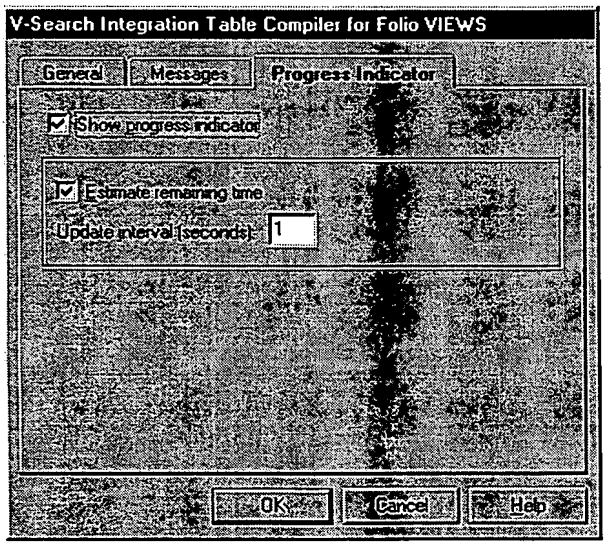
/Warnings=[yes|no]
/ReportTo=[dialog|file]
/LogFile=*filename*
/LogOption=[append|overwrite]



Progress Indicator Tab

Use the Progress Indicator tab to enter these options:

/Progress=[yes|no]
/EstimateTime=[yes|no]
/UpdateInterval=[1| *seconds-between-updates*]



Parameter Summary

The parameters for the integration table compiler (VSFVCOMP) are summarized in the following table:

Parameter	*Required?*	*Contains*	*Default Value*
ParamFile	No	Name of parameter file	none
Database	Yes	Name of VDB file	none
Table	Yes	Name of the Folio VIEWS Integration Table	none
Warnings	No	yes | no	yes
ReportTo	No	dialog | file	dialog
LogFile	No	Name of a log file.	Integration Table Compiler Log.txt
LogOption	No	append, overwrite	append
Progress	No	yes | no	yes
EstimateTime	No	yes | no	yes
UpdateInterval	No	Number of seconds between updates (0-60) 0 means update as fast as possible.	1
StartupDialog	No	yes | no	yes (if no VDB file has been specified)
Target	No	Name for the compiled Folio VIEWS Integration file.	Path is the same as the path for the VDB file. The file name is the same as the table name with an FTB extension.
Overwrite	No	yes | no	no

Exit Codes

The compiler returns an exit code that indicates whether errors have occurred. The exit code can be used in DOS batch files to determine whether to continue with other commands in the batch file. An exit code of zero (0) indicates that there is no error. Any exit code greater than 0 indicates an error. When you get an error exit code, look at the message in the dialog box or in the log file for detailed error information.

Chapter 4: Creating Jumps From Folio VIEWS to V-Search

This chapter describes how to create jumps from VIEWS text to a V-Search Map by adding program links to the Folio flat files. When you add these program links, your users will be able to click on a program link in the text to display a V-Search Map for that node.

For each program link you want to add to the text of a document, code a Program Link tag in the Folio flat file. The Program Link calls the V-Search program VSFVJUMP.EXE and passes the NodeID of a node in the Node table. VSFVJUMP uses the NodeID to display a search dialog box (see Figure 5) for that NodeID showing the available maps for that node.

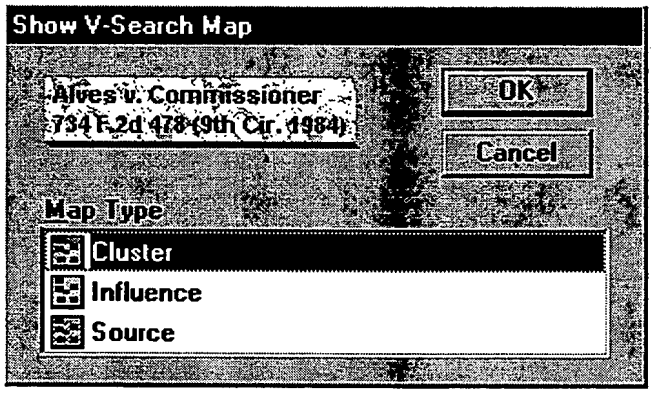


Figure 5. Search Dialog Displayed by VSFVJUMP.

Why Use a Bitmap ?

At Libertech, we found that we could use either highlighted text or a small bitmap as a hotspot to launch a V-Search Map. However, we also use text hotspots as Jump Links within the VIEWS text. It seems simpler for the end user if we use something distinctly different to launch the maps, so we use a small bitmap.

Two Kinds of Bitmaps?

We use the program-link bitmaps in two different ways in the VIEWS text.

- One kind of bitmap, called a ***node bitmap***, marks the start of the text represented by a node. When an end user clicks on one of these, a search dialog is displayed for the current node.
- Another kind of bitmap, called a ***link bitmap***, marks a reference from the text of one document to other documents. When an end user clicks on a link bitmap, a search dialog is displayed for the node being cited from the current document.

When you install the V-Search Integration Toolkit for Folio VIEWS, we provide these two bitmaps as bitmap files named NODE.BMP and LINK.BMP.

Creating a Program Link in the Flat File Text

At each place in the infobase text where you want to imbed a program link, add a Program Link tag that looks like this:

```
<PL:style_name, "VSFVJUMP command ">
<OB: "bitmap_file_name ", Bitmap>
</PL>
```

Where you replace:

Item	Meaning
style_name	With the Folio VIEWS style name you want to assign to the bitmap named in the Program Link tag. You must also define this style name in the VIEWS *.DEF flat file.
VSFVJump Command	With a VSFVJump command, as shown below.
bitmap_file_name	With the name of a bitmap file (*.BMP) to display as the hotspot. You must also define this bitmap in the VIEWS *.DEF flat file. Use an Object Definition tag and a class name of Bitmap.

VSFVJump Command

Use this syntax for the VSFVJump command in a VIEWS program jump.

VSFVJUMP [[Search=]*NodeID*[/Map=*LinkTypeName*]]

Where you replace:

Item	Meaning
NodeID	With the NodeID of the node whose maps you want to generate.
LinkTypeName	With the LinkTypeName that specifies the kind of map to display at startup.

You should enclose the options in double-quotes if they can be interpreted ambiguously. You can specify the VSFVJump options in one of these ways:

1. Specify just the NodeID:

   VSFVJump 91 T.C. 200

2. Specify /Search and the NodeID:

   VSFVJump /Search=91 T.C. 200

3. Specify /Search and the NodeID, with /Map:

   VSFVJump /Search="91 T.C. 200" /Map="Cluster"

Providing a Startup Map

You can use the /Search=*NodeID* and /Map=*LinkTypeName* options to start the viewer at a specified map.

1. If the /Search=*NodeId* and the /Map=*LinkTypeName* are both valid, the specified map is displayed for the NodeID. The user will see two VIEWS windows: the infobase text in one, and the V-Search map in the other.
2. If the /Search=*NodeId* is specified but is not found in the database, an error message is displayed. Neither a map nor a dialog is displayed to the user.
3. If the /Search=*NodeId* is valid, but the /Map=*LinkTypeName* is missing, invalid, or unavailable for the *NodeID*, an error message is displayed. Neither a map nor a dialog is displayed to the user.
4. If the /Search=*NodeId* is not specified, neither a startup map nor a dialog are displayed.

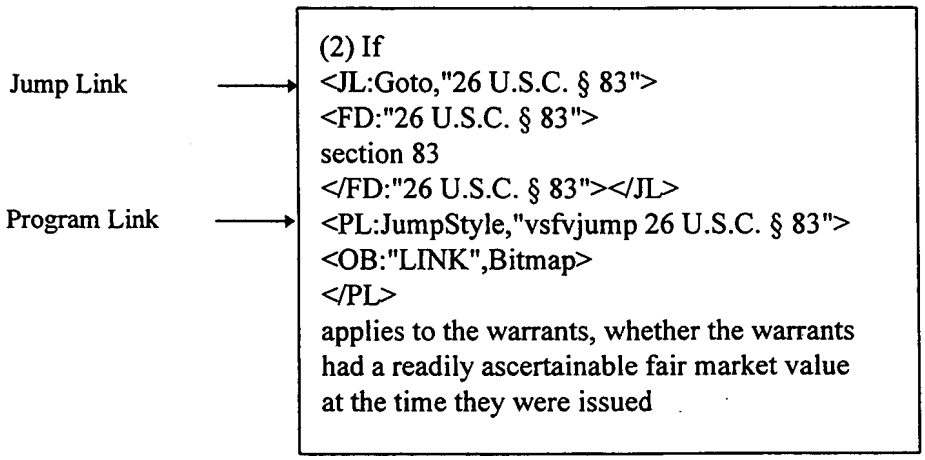
Sample Program Link Tag

Here is a sample Program Link tag:

```
<PL:JumpStyle,"vsfvjump 26 U.S.C. § 83"><OB:"LINK",Bitmap></PL>
```

Sample Program Link Tag in Text

Here is a more complete example of a program link set in infobase text. Note that we have added the Program Link tag after a Jump Link tag. This infobase uses Jump Link tags for citations to other text documents in the infobase. When the text is marked for citations, it can also be marked for Program Links. Often the same tool that adds the Jump Link tags to a document can be modified to add the Program Links as well.



Chapter 5: Marking Up Your Folio Flat Files for V-Search

This chapter describes some methods you can use to modify your Folio flat files for use with V-Search. You are probably already using some set of Folio flat file tags to mark your current text files before converting them to infobase format. For example, perhaps you already use Jump Link tags to create hyperlinks from one document to another. In this chapter, you will find ways to use new tags, and ways to modify the tags you may already be using. In particular, you will be concerned with ways to add or change Folio flat files to:

- Add Program Link tags that jump from the text to V-Search Maps.
- Provide entry points for Show Text commands that jump from the maps into the text of a document.
- Provide entry points for Show Usage commands that show references in one document to another.

Markup for Program Links from Text to Maps

To provide a hotspot for the user to click on to jump from VIEWS text to a V-Search map, you insert a Program Link tag in the VIEWS text, as described in "Chapter 4: Creating Jumps From Folio VIEWS to V-Search" on page 20. In this sample, the text is originally marked with Jump Link (<JL>) tags to create hyperlinks within the text of the VIEWS document. For example, if you already use Jump Link tags for hyperlinks like this:

```
(2) If
<JL:Goto,"26 U.S.C. § 83">
<FD:"26 U.S.C. § 83">
section 83
</FD:"26 U.S.C. § 83"></JL>
applies to the warrants, whether the warrants
```

Then you can insert a Program Link tag right after the Jump Link tag to create a jump to a V-Search Map:

```
(2) If
<JL:Goto,"26 U.S.C. § 83">
<FD:"26 U.S.C. § 83">
section 83
</FD:"26 U.S.C. § 83"></JL>
<PL:JumpStyle,"vsfvjump 26 U.S.C. § 83">
<OB:"LINK",Bitmap>
</PL>
applies to the warrants, whether the warrants
```

Markup for Show Text Commands from Maps to Text

To implement the Show Text command from a V-Search Map, you need some way of delimiting the start of the text to be viewed for each NodeID. You can do this in several ways. For example, you may already use a Jump Destination (<JD>) tag to mark the entry point to a document for Jump Links from other documents. Or, you may already use a Group <GR> tag to mark all the records in a document.

Use the technique that works best for your end users. Be sure to read "Displaying Text with Show Text and Show Usage " on page 10 before you select your method.

Using Jump Destination Tags

If your text is marked up with Jump Destination <JD> tags, you can use Jump commands to jump from V-Search Maps to the VIEWS text. Here is some sample text:

```
<JD:26 U.S.C. § 83>
  .
Text
  .
```

And here is a slice from an integration table:

NodeID	Text Length	Show Text Method	ShowTextCmd	ShowUsageQuery
26 U.S.C. § 83	2343	Jump	#scope#	

Note that the ShowTextCmd column uses the scope symbol to specify a NodeID as the name of a Jump Destination for the VIEWS Jump command in the ShowTextMethod column.

Using Group Tags

If your text is marked up with Group <GR> tags, you can use Query commands to jump from V-Search Maps to the VIEWS text. Here is the sample text:

```
<GR:92 T.C. 612>
  .
Text
  .
```

And here is an entry from the integration table:

NodeID	Text Length	Show Text Method	ShowTextCmd	ShowUsageQuery
92 T.C. 612	4098	Query	[group "#scope#":*]	

The ShowTextCmd column contains a Query for the group with the name specified by the NodeID in the scope symbol.

Markup for Show Usage Commands from Maps to Text Using Field Tags

If you use both Group and Field tags in your VIEWS markup, you can easily implement the Show Usage command. Use the:

- Group tags to mark the records that contain the text of a node. Each record in the text must include a <GR> tag.
- Field tags to mark references, or citations, to other nodes from this node.

Here is the sample text:

```
<GR:"92 T.C. 612">
(1) Whether the stock warrants were issued "in connection with
the
performance of services" within the meaning of
<FD:"26 U.S.C. § 83">
section 83
</FD:"26 U.S.C. § 83">
```

Notice the <GR> tag at the beginning of the text that indicates that this record belongs to document 92 T.C. 612. This document refers to another document, 26 U.S.C. § 83. The <FD> Field tag marks the reference to 26 U.S.C. § 83. The <JL> tag provides the VIEWS hyperlink to the text of 26 U.S.C. § 83.

Here is the sample integration table for Show Usage of 26 U.S.C. § 83:

NodeID	Text Length	Show Text Method	ShowTextCmd	ShowUsageQuery
26 U.S.C. § 83	2343	Jump	#scope#	[group "#scope#": [field "#reference#": *]]

This ShowUsageQuery tells VIEWS to:

- Find all the records in the group whose ID is the #scope# NodeID;
- That contain Field tags that have references to the #reference# NodeID;
- And display the text beginning with the first record.

Chapter 6: Completing the Package

This chapter describes what you need to include when you ship an integrated V-Search database with your VIEWS infobases on a CD-ROM. It also includes a description of the interaction between the VIEWS Infobase Manager and the V-Search Viewer.

Files to Include

Be sure to include these files on the CD-ROM:

File	Purpose	Install Where
VSRCHFVE.DLL	Is the 16-bit V-Search Extension to Folio VIEWS. This library provides for displaying V-Search Maps from a Folio VIEWS Infobase Manager.	Any directory; referred to in VIEWS.INI.
FOLIOVWS.VXT	Is the 16-bit Folio VIEWS V-Search Extension Library. This library provides for displaying Folio VIEWS text from a V-Search Map.	Same directory as VSRCHFVE.DLL
VSFVJUMP.EXE	Displays V-Search Maps from program links in the infobase text.	Path and directory accessible to VIEWS.
infobasename.FTB	Compiled Folio VIEWS Integration Tables; one for each infobase, with same filename as the infobase name.	Same directory as infobase.
VSEARCH.VVW	Compiled V-Search Database. This viewer file is compiled from the VDB file, and you must change the name to VSEARCH.VVW.	Same directory as infobase.

Modifying the VIEWS.INI File

To run V-Search Maps from VIEWS, you must change the VIEWS.INI file. In the [AddOns] section of this file, add the path and name of the V-Search viewer. If the [AddOns] section does not exist, you can add one.

```
[AddOns]
Libertech V-Search=d:\libertec\bin\vsrchfve.dll
```

Modifying the Database Profile (VDB) File

The database profile (VDB) file contains some entries that you modify to suit your own database. Then you compile the VDB file to get a V-Search Viewer (VVW) File for use with the viewer.

Before you Compile the VDB File

Before you compile the VDB file, make these changes to the [Database Properties] section.

Add the LoadFolioIntTables Entry

Add an entry to the [Database Properties] section of the VDB file to list all of the compiled Folio VIEWS Integration Tables associated with this V-Search database. The entry looks like this:

```
[Database Properties]
LoadFolioIntTables=\path\tablename1, \path\tablename2, ... \path\tablenameN
```

Where you replace:

path With the optional path to the table.

tablename With the name of a compiled Folio VIEWS Integration Table (which must have a file extension of FTB).

Change the HelpComand Entry

If you want to provide your own help for your database, change the HelpCommand entry in the [Database Properties] section of the VDB file. The HelpCommand entry contains the actual Windows command-line command used to display help for the database. The initial VDB file has a HelpCommand entry with no actual command. The entry looks like this:

```
[Database Properties]
HelpCommand=
```

If you leave this entry as is, and do not supply your own online help, the viewer will use the help file provided with V-Search. V-Search ships two sample Help files to accompany the sample legal database. They are:

VSEARCH.HLP	The Help file in WinHelp format.
VSEARCH.NFO	The Help file in Infobase format.

If you want to provide your own WinHelp online help, modify the [Database Properties] section like this:

```
[Database Properties]
HelpCommand= MyHelp.HLP
```

If you want to provide your own online help and your own online help command, modify the [Database Properties] section like this:

```
[Database Properties]
HelpCommand= MyHelpCommand MyHelp.dat
```

Change the StartupSearch and StartupMap Entries

If you want to have the V-Search viewer start by showing a specific map for a selected node, change the StartupSearch and StartupMap entries in the [Database Properties] section of the VDB file. The entries looks like this:

```
[Database Properties]
StartupSearch=
StartupMap=
```

You can add a NodeID and LinkTypeName to start the viewer. For example, to start the viewer with a Cluster Map for node 91 T.C. 200, you would enter:

```
[Database Properties]
StartupSearch=91 T.C. 200
StartupMap=Cluster
```

How it Works

When the viewer starts, it first looks at the StartupSearch and StartupMap entries:

1. If the StartupSearch=*NodeId* and the StartupMap=*LinkTypeName* are both valid, the specified map is displayed for the NodeID. The user will see two VIEWS windows: the infobase text in one, and the V-Search map in the other.

2. If the StartupSearch=*NodeId* and StartupMap=*LinkTypeName* are not specified, is invalid, or is not found in the database, a startup map is not displayed; the user sees only the infobase text.

When you Compile the VDB File

When you compile the database profile (VDB) file, set the /Target option to:

/Target=VSEARCH

The V-Search Viewer for Folio VIEWS expects to find VSEARCH.VVW as the database name. You can either set it in the /Table option of VSDBCOMP, or change it after you compile. In any case, the V-Search Viewer for Folio VIEWS will open only VSEARCH.VVW.

Providing Online Help for VSRCHFVE.DLL

V-Search provides two kinds of online help for the viewer. There is online help for the viewer (VSRCHFVE.DLL) itself, and there is help for the database to be viewed. V-Search provides:

Help File	Purpose
VSRCHFVE.HLP	Online help for the viewer in WinHelp format.
VSRCHFVE.NFO	Online help for the viewer in VIEWS Infobase format.
VSEARCH.HLP	Online help for the sample legal database (LGLSAMP01) in WinHelp format.
VSEARCH.NFO	Online help for the sample legal database (LGLSAMP01) in VIEWS Infobase format.

Viewer Help

When the user selects *Help on V-Search* from the menu bar, the viewer:

1. First looks for the file VSEARCH.NFO in the same directory with VSRCHFVE.DLL and uses VIEWS to display the help. V-Search provides the VSEARCH.NFO infobase.
2. If VSEARCH.NFO is unavailable, the viewer looks for the file VSEARCH.HLP in the same directory with VSRCHFVE.DLL and uses WinHelp to display that help. V-Search provides the VSEARCH.HLP help file.
3. If neither is available, Help is not available for the viewer.

Be sure to include either VSEARCH.HLP or VSEARCH.NFO with the viewer when you package the viewer with your infobases for delivery to end users.

Database Help

When the user selects *Help on the Database* from the menu bar, the viewer:

1. Looks in the [Database Properties] section of the VVW file for a Help command to execute. If a command is there, the viewer runs it.
2. If there is no Help command in [Database Properties], the viewer looks for VSEARCH.NFO in the same directory as the V-Search database, and runs it using VIEWS.
3. If there is no VSEARCH.NFO, the viewer looks for VSEARCH.HLP in the same directory as the V-Search database, and runs it using WinHelp.
4. If the viewer cannot find online help, then Help is unavailable.

Providing Custom Database Help

If you want to provide your own online help for your infobases, you can create your own help information, and even the command to display it. Then add the help file, and new command if you have one, to the HelpCommand entry in the [Database Properties] section of the VDB file. See "Change the HelpComand Entry" on page 27 for more information.

What Happens When You Integrate Folio VIEWS with VSRCHFVE.DLL

This section describes how the V-Search Viewer for Folio VIEWS (VSRCHFVE.DLL) interacts with the Folio VIEWS Infobase Manager to integrate text and maps.

What You Need ...

Before startup can take place, you must have:

- All the Folio VIEWS infobases (*filename*.NFO) to be viewed by the end user.
- One compiled V-Search Database named VSEARCH.VVW. This is the output of the V-Search Publisher's Toolkit Database Compiler (VSDBCOMP), as described in the Publisher's Toolkit User's Manual. The V-Search database has properties that list the names of, and paths to, all the V-Search Folio VIEWS Integration Tables. (The VSEARCH.VVW file must be in the same directory as the NFO files.)
- One compiled V-Search Folio VIEWS Integration Table (*filename*.FTB) for each infobase file. Each integration table must have the same filename as the infobase with which it is associated. The FTB files associate nodes in the V-Search database with VIEWS commands.
- One modified VIEWS.INI file with this entry added:

```
[AddOns]
Libertech V-Search=drive:\directory\vsrchfve.dll
```

- These V-Search components:
  - VSRCHFVE.DLL, the V-Search Viewer, which is an extension to VIEWS.
  - FOLIOVWS.VXT, the 16-bit V-Search Extension Library for Folio VIEWS, which is used by VSRCHFVE.DLL.

How it All Works ...

After everything is installed, the Folio VIEWS Infobase Manager and the V-Search Viewer work together to display both the text and the maps, as shown in Figure 6.

1. The end user starts the Folio VIEWS Infobase Manager.
2. The Infobase Manager looks in its VIEWS.INI file for any extensions to load. Extensions are specified in the [AddOns] section.
3. The Infobase Manager finds the Libertech V-Search entry in the [AddOns] section. It adds V-Search to the VIEWS menu bar and starts the V-Search Folio VIEWS Extension (VSRCHFVE.DLL), which is the V-Search Viewer.
4. The V-Search Viewer looks in its directory for any 16-bit V-Search extension libraries (with a file extension of VXT) and *loads* the FOLIOVWS.VXT extension.

   Note: At this point, neither an infobase nor a V-Search database has been opened.

5. The end user uses VIEWS to open an infobase.
6. VSRCHFVE.DLL *starts* its Folio VIEWS extension (FOLIOVWS.VXT).
7. FOLIOVWS.VXT finds the database properties, and opens the compiled Folio VIEWS Integration Tables. There should be one FTB file for each NFO infobase file.
8. VSRCHFVE.DLL opens VSEARCH.VVW (the V-Search database). After the VVW files are open, the end user can jump from the VIEWS infobase text to a V-Search map, and back to VIEWS infobase text.

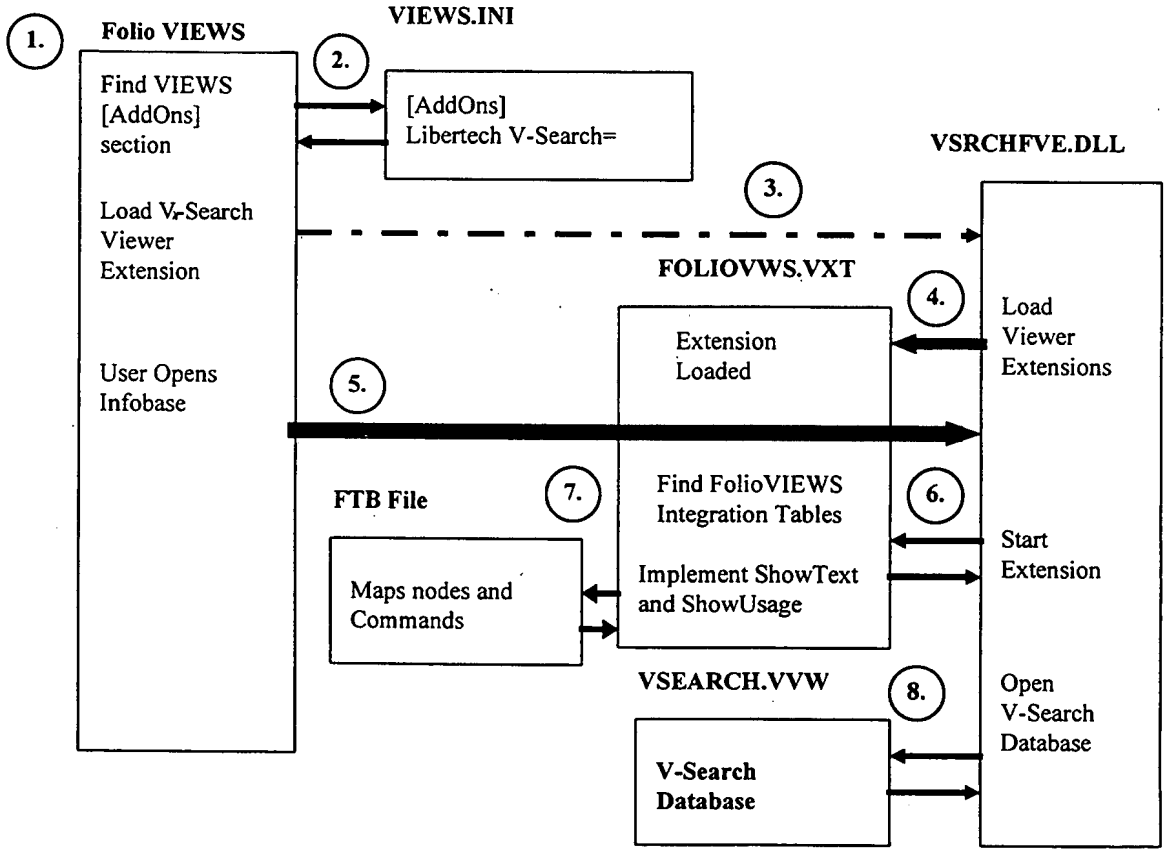


Figure 6. Folio VIEWS and V-Search Viewer Startup.

Glossary

This glossary defines the terms used throughout this manual.

AliasID: An identification of an element in the integration table that contains a generic VIEWS command that can be referred to by the NodeID entries in the table.

Cluster link: A relationship between two nodes based, by default, on a statistical analysis of multiple relationships between nodes in a database. For example, two nodes, both directly linked to the same intermediate nodes, may be indirectly linked through many paths and therefore have a Cluster link between them.

Cluster map: A V-Search Map that shows Cluster links from the search node.

compiled database: A compact and efficient version of the V-Search database suitable for CD-ROM publishing. The compiled V-Search database is a V-Search Viewer file with a file extension of VVW.

Folio flat file: An ASCII representation of a Folio infobase with tags to control text formatting.

icon: A small graphic or picture that represents a minimized, running Windows application or V-Search Map. For the V-Search Viewer, icons represent active maps that have been minimized in the V-Search application window.

Influence link: A relationship between two nodes such that one node is *referred to* by another node. The node being referred to is said to have an Influence link to the node making the reference.

Influence map: A V-Search Map that shows all nodes joined by Influence links from the search node.

link: A relationship, either stated or implied, between two nodes. A link is represented by an entry in the Link table in the V-Search database that contains a FromNodeID, a ToNodeID, a link type, and a Weight.

map: The graphical output of a V-Search search. A map shows all nodes linked to the search node by the same type of link.

node: Conceptually, any entity that can be represented by a box on the V-Search Map, and that has links to other nodes. Practically, a node might be a document, a section of a document, or a concept, such as a topic name, that has sources and influences. A node is defined as something that has a unique NodeID, a node type, a node subtype, and a plot date.

node box: The graphic representation of a node on a V-Search Map. A node box is a rectangle, often showing a bottom surface and a right edge. The face displays the NodeID of the node and any optional comments. By convention, the thickness of the box represents the relative size of the underlying document.

NodeID: The unique identification given to an entry in the Node table. A NodeID usually identifies a document or a section of a document in an infobase.

ODBC: Open Database Connectivity. ODBC is the name of a standard for accessing information in a relational database.

search node: The node displayed on the node box in the search area. This is the node that is the search argument, the node for which a set of nodes joined to it by links of a particular type are displayed in the result area, and the focus of a V-Search Map.

source link: A relationship between two nodes such that one node *refers to* the other node. The node that refers to the other node has a Source link to that other node. The reference can be an actual stated reference, or an implied reference.

Source map: A V-Search Map that shows Source links from the search node.

to-node: The node to which a link is made. For a Source link, the to-node is the node that is referred to. For an Influence link, the to-node is the node that made the reference.

VDB files: Database profile files, with the file extension VDB, that describe V-Search databases.

VSDBEDIT: The V-Search Database Editor. VSDBEDIT is a part of the Publisher's Toolkit that lets you add and edit tables in the uncompiled V-Search database.

V-Search database: The user-accessible, editable, plain text set of database tables before they are processed by the V-Search database compiler. You can use a database manager to view and edit the information in the V-Search database before it is compiled.

VSRCHFVE.DLL: The V-Search Viewer for Folio VIEWS. VSRCHFVE.DLL is part of the V-Search Integration Toolkit for Folio VIEWS and runs as an extension to the VIEWS Infobase Manager.

Index

—W—